CTF 领域指南

wizardforcel

Published with GitBook

目錄

介绍	0
一、夺旗比赛	1
1.1、找一场CTF	1.1
1.2、找一份工作	1.2
二、漏洞挖掘	2
2.1、源代码审计	2.1
2.2、二进制审计	2.2
2.3、Web应用审计	2.3
三、漏洞利用程序的构建	3
3.1、二进制的漏洞利用（1）	3.1
3.2、二进制的漏洞利用（2）	3.2
3.3、Web应用的漏洞利用	3.3
四、电子取证	4
五、工具箱的构建	5
5.1、工具箱的准备	5.1
六、可实施的谍报技术	6
6.1、案例研究	6.1
七、参与者	7

CTF 领域指南

原文：CTF Field Guide

译者：做个好人

来源：CTF领域指南

协议：CC BY-NC-ND 3.0

译者声明：本文翻译自美国trailofbits团队发布在Github上的《CTF Field Guide》手册，我们已与对方联系获得翻译授权，由于手册内容尚在更新过程中，故有部分缺失。如有翻译不当之处，请与我们联系更正：@IDF实验室。

> “Knowing is not enough; we must apply. Willing is not enough; we must do.”（仅仅知道还不够，我们必须付诸实践。仅有意愿还不够，我们必须付诸行动。）
>
> —— Johann Wolfgang von Goethe

欢迎！

很高兴你能来到这里，我们需要更多的像你这样的人。

如果你想拥有一种能够自我防卫的生活，那就必须像攻击者那样思考。

所以，学学如何在夺旗比赛（CTF）中赢得比赛吧，这种比赛将专业的计算机安全活动规则进行了浓缩，且具有可客观评判的挑战题。CTF比赛趋向关注的主要领域是漏洞挖掘、漏洞利用程序构建、工具箱构建和可实施的谍报技术（tradecraft）。

无论你想赢得CTF比赛，还是想成为一名计算机安全专家，都至少需要擅长这些方面中的其中之一，理想情况下需要擅长所有这些领域。

这就是我们编写此手册的目的。

在这些章节中，你会找到赢得下一次CTF比赛的一切要素：

- 以往CTF挑战赛的题目的查看和研究；
- 帮助你设计和构建属于自己工具集的引导；
- 包括现实世界中的和以往CTF比赛中的攻击者行为案例研究；

为了让你更好过一些，我们在每节课程中补充了互联网上最好的配套参考资料，这些资料来自于一些计算机安全领域最好的人员之手。更进一步，我们希望你能够将此手册与实际工作结合起来使用。

我们编写该手册是为了帮助你能够尽可能地快速学习，如果在学习过程中有任何问题，请与我们联系，我们将把你的问题转达给最合适的专家。如果有足够的需要，我们甚至可以安排一节线上课程。

现在，让我们开始吧。

一、夺旗比赛

为什么选择**CTF**？

计算机安全因其跨领域的特性在人才教育方面提出了一个挑战。计算机安全的课题领域的分布从计算机科学理论方面到信息技术管理方面的应用，这样很难简短概括计算机安全专业的灵魂是什么。

评估这种专业性的一种近似方法已经出现："夺旗"比赛。攻击为导向的CTF比赛尝试将专业计算机安全工作许多方面的本质浓缩为可客观评估的简短挑战题目。CTF比赛趋向于关注的领域包括漏洞挖掘、漏洞利用程序构建、工具箱构建和可实施的谍报技术（tradecraft）。

一个现代的计算机安全专家应该至少其中一个领域的专家，理想情况下应该擅长所有这些内容。在CTF比赛中赢得胜利要求参赛选手在所有这些领域中至少是其中一方面的专家。所以，准备和参与CTF比赛是一种有效将计算机科学的离散面聚合、聚焦于计算机安全领域的方法。

1.1、找一场CTF

如果曾经你想开始练习跑步，那么可能你是将5000米作为一个目标。同样的原则在这里也适用：选择一个在不久之后你想要参加的一场CTF比赛，并设定一个练习计划。下面是我们推荐的一些CTF比赛：

- CMU主办的PicoCTF和PlaidCTF
- 针对高校学生的HSCTF
- Ghost in the Shellcode（Gits）
- NYU-Poly主办的CSAW CTF
- 只针对学院派的UCSB iCTF
- Defcon CTF

访问CTF Time和CapCTF calendar可以看到一年中每个星期都在发生的更多、更全的CTF比赛。

Wargame有什么不同？

除了持续进行之外，Wargame和CTF很相似。典型的情况是，Wargame题目被组织为许多级别，并随着被解决题目的增多而逐渐变难。Wargame是为CTF进行练习的绝佳方式！以下是我们最喜欢的一些Wargame站点：

- Micro Corruption
- SmashTheStack
- OverTheWire
- Exploit Exercises

CCDC怎么样？

有一些仅做防御的比赛也将自己看作是CTF比赛，主要是高校网络防御挑战赛（CCDC，Collegiate Cyber Defense Challenge）及其地区比赛，我们的建议是你应该无视这些比赛。无奈的是它们的题目都是不切实际的题目，很少会教授你和安全相关的内容。虽然他们乐此不疲的将自己作为红队（Red Team）看待！

1.2、找一份工作

职业列表

[编辑注：这是为pentest.cryptocity.net编写的一篇较老的文章，我们正在更新过程中。]

以下这些是基于我已有的经验和你情况的不同而写的对于信息安全职业生涯的看法。如果你住在纽约市，而且对于应用安全、渗透测试或逆向工程感兴趣，刚刚踏入信息安全领域开始你的职业生涯，那么以下信息会对你非常适合。

1. 雇主
2. 角色
3. 书本中学习
4. 课程中学习
5. 大学
6. CTF比赛
7. 沟通
8. 见人
9. 会议
10. 认证
11. 链接
12. 指引的朋友

雇主

就我可讲的而言，在信息安全产业有五类主要的雇主（不计学院派）。

- 政府
- 非技术型财富500强（大多数是金融类）
- 大型技术供应商（大多数是在西海岸）
- 大型咨询公司（大多数非技术企业）
- 小型咨询公司（大多数很酷）

你所工作的产业决定了你需要解决的主要问题。例如，金融行业重点强调的是对于商业过程将风险降低到最低损失（大规模自动化的原因）。另一方面，咨询常常意味着向人们销售想法X，X实际上就是一个漏洞或研究发现新的漏洞。

角色

我主要将信息安全职位分离对应到互联网网络安全、产品安全和咨询。进一步我将这些类型的工作分类为以下几个角色：

- 应用安全（代码审计/应用评估）
- 攻击者（攻击）
- 合规性
- 电子取证
- 事件处理
- 管理者
- 网络安全工程师
- 渗透测试
- 政策/策略
- 研究员
- 逆向工程师
- 安全架构师

以上的每一个角色都要求有不同的、高专业性的知识面。这个站点对于应用安全和渗透测试是一个不错的资源，但是如果你对其他角色感兴趣就需要找其他资源。

书本中学习

幸运的是，关于每个信息安全方面的主题都有一打好书。Dino Dai Zovi和Tom Ptacek都有绝佳的读书列表。我们建议阅读以下图书：

- Gray Hat Hacking
- The Myths of Security
- Hacking: The Next Generation
- O'Reilly出版的任何一本你选择的脚本语言书

如果你不知道自己在找什么书，那么可以访问O'Reilly提供的书单。他们可能是这个产业中最执着和最高质的图书出版商了。

别忘了，单纯的读书不会给予你超越谈资之外的任何附加技能，你需要练习或基于你从书本中实际学到和理解的内容创造一些东西。

课程中学习

如果你在找一些可以随手拿来和直接了当的东西，那么有许多在线的大学课程是关于信息安全的。我在下方列出了一些绝佳的课程资源（根据机构名称排序）。RPI课程是和这些课程最相似的一个，Hovav可以通过最佳学术阅读列表的内容获得绩点，但是列表中的每个课程都很出色。

[编辑注：表格待添加/之后更新。]

大学

找到一所有专业安全课程大学的最容易和简单的办法是通过NSA Centers of Academic Execllence（NSA-COE）机构列表。这个资质要求正逐渐放低以至于越来越多的大学已经获得此资质，它也可以帮助你寻找那些刚刚获得COE-CO资质的单位。记住，资质仅供参考。你需要深入了解每所大学实际的课程，而不是仅依靠资质做选择。

一旦进入大学，要上那些能够强迫你编写大量代码解决难题的课程。IMHO的课程聚焦于理论或提供有限价值的模拟问题。如果你无法从整个CS课程表中确定哪些是有编程内容的CS课程，那么就向学长征求意见。另一种达成此项目的的方式是报考学校的软件开发专业而不是计算机科学专业。

夺旗比赛

如果你想获得和学到技能技巧，且想要做地更快，那么应该参加CTF比赛或一头扎进Wargame。值得注意的一点是许多这种挑战赛都有附加的会议（各种规模），参加这些比赛就意味着会错过整个会议。试着不要赛过头，因为参加会议有参加会议的好处（见下文）。

有一些仅做防御的比赛也将自己定位为CTF比赛，主要是高校网络防御挑战赛（CCDC）及其地区比赛，我的建议是无视它们。他们的题目是关于系统管理，很少会教你安全相关的内容。尽管他们乐此不疲地将自己看作是红队（Red Team）。

沟通

在任何角色中，你的时间都主要花费在与其他人的沟通上，主要通过Email和会议，少量是通过电话和即时通讯（IM）。雇主的角色决定了你是否需要和内部安全团队、非安全技术人员或商业用户接触更多。例如，如果你在金融公司做网络安全工作，那么就需要和内部技术人员沟通更多。

在大型组织中做好沟通的建议：

- 学习写简洁、明了、专业的邮件。
- 学习让事情有组织地被解决。不要随心所欲。

- 学习公司或客户的商业内容。如果你从商业层面考虑，你的选择就有a）不做任何事 b）解决事情 和c）考虑时间和成本更有说服力做事。
- 学习你所在公司或客户是怎样工作的，比如关键人物、过程或其他能够让事情达成的激励因素。

如果你仍然在学校学习CS课程，参加人文类课程会强迫你写东西。

见人

找到并参加你所在城市的安全聚会（CitySec），在大多数城市每个月都会有一次没有演讲的非正式会面。顺道提一句，我们参加的是我们本地的NYSEC。

ISSA和ISC2关注于政策、合规性和其他新兴且不确定的话题。类似的，InfraGard主要关注在非技术性法律执行方面的话题。OWASP是由厂商发起形成的活动中最糟糕的例子，与其说它是和技术相关，不如说是和销售相关。

会议

如果你此前从来没参加过信息安全会议，使用下面的Google日历可以帮你找到一个低消费的本地会议。我的学生中有人认为参加一场会议是某种测验，所以尽可能地推脱不去。我保证我不会带着期末测验从灌木丛中跳出来，并在事后扣你的分数。

- 信息安全会议日历

如果你参加一场会议，不要太看重每个时间段的会谈。会谈只是吸引所有聪明的黑客在某个周末聚集到一个角落的诱饵：你应该见见其他与会者！如果一个特别的会谈非常有意思和有用，那么你应该和演讲者进行互动。关于这个主题，Shawn Moyer在Defcon嘉宾角的这篇文章中有更多描述。

如果你在某个地方工作，并焦虑着如何向公司交代出席会议的事情，那么Infosec Leaders blog中有一些有用的建议。

认证

这个产业需要特别的知识和技巧，为了认证考试而学习不会对你的知识和技能有任何帮助。事实上，在很多情况下它还有坏处，因为你花费时间学习认证考试而导致你分心无法做该手册中提到的事情。

即便如此，有一些便宜的和中立厂商的认证可以在你当前阶段帮助你突出你的简历，例如Network+和Security+或者甚至一个NOP，但是我认为认证在你找工作或专业发展中起不到什么作用。

通常，有两种原因需要获得认证：

- 你已经支付认证的费用，通过支付培训和考试或有时候在你获得认证之后会自动支付费用（通常是政府认证）。

- 你的公司或你的客户强迫你获得认证。这通常有助于销售增长，比如“你应该选择我们，因为我们所有的员工都有XYZ认证！”

通常，花费时间在参加CTF上更加有产出效率，然后用你的最终排名来证明你的能力。

链接

- 关于该文章的Reddit和Hacker News子内容
- 安全建议
- How to Break Into Security, Ptacek Edition
- VRT: How to Become an Infosec Expert, Part I
- Five pieces of advice for those new to the infosec industry
- How to Milk a Computer Science Education for Offensive Security Skills
- Kill Your Idols, Shawn Moyer's reflections on his first years at Defcon
- 认证有感
- My Canons of (ISC)2 Ethics
- Not a CISSP
- (ISC)2's Newest Cash Cow
- Why You Should Not Get a CISSP
- 常规技术建议
- Advice for Computer Science College Students
- Don't call yourself a programmer, and other career advice
- The answer to "Will you mentor me?" is …. no.

二、漏洞挖掘

2.1、源代码审计

这个部分是关于熟悉应用程序编译为本地代码时显现的漏洞。对一门编译语言编写应用程序时的精准和完整理解，在没有学习编译器怎样转换源代码为机器语言和处理器怎么执行代码前是无法达到的。一种简单的获得这些转换经验的方式是通过逆向工程你自己的代码或源码可见的项目。在这个部分结束时你将会识别用诸如C和C++编译语言编写的常见漏洞。

大型软件包由于使用第三方软件库导致漏洞普遍存在。常见的例子包括像libxml、libpng、libpoppler和用来解析已编译文件格式和协议的libfreetype等这样的库。这些库中的每一个都曾在历史上被发现过易于攻击的漏洞。即便当新的版本发布之后，也无助于绝大多数软件的未更新，在这些情况下很容易发现易于发现的已知漏洞。

课程

- Source Code Auditing I
- Source Code Auditing II

挑战工坊

为了锻炼你的技能，我们建议通过一个故意留有漏洞的程序过一遍尽可能多漏洞发掘之旅，然后转移到实际应用中做同样的事情。

Newspaper应用是一个用C语言编写的小型服务应用，允许认证的用户读取和写入文章到一个远程文件系统。Newspaper编写的方式使得它对于许多不同的攻击都有漏洞可以利用。你应该能够通过源码发现其中至少10个bug或可能存在的漏洞。

- Newspaper App
- Newspaper App Installer

Wireshark，无论怎样，是一款从1998年开始持续开发的行业标准级别的网络协议分析器。相比漏洞诸多的Newspaper应用，Wireshark的漏洞少之又少。查看wireshark安全页面，找到一个协议解析器的名字并测试是否你可以在没有查看漏洞细节的情况下发现漏洞。解析器位于/epan/dissectors/目录。

- Wiireesakk 1.8.5

工具

当进行源码审计时，使用一款用户分析和引导代码库的工具是很有帮助的。下面是两款工具，Source Navigator和Understand，通过收集和展现相关数据关系、API使用、设计模式和控制流的信息来帮助分析员快速熟悉代码。一个有用的对比工具的例子同样列在下方。其中一个免费、开源的审计工具例子是Clang Static Analyzer，该工具可以帮助你在常见API和漏洞编码形式中跟踪编程错误。

- Source Navigator
- Scitools Understand
- Meld
- Clang Static Analyzer

资源

要确定你对分析目标所用的编程语言非常地熟悉。发现漏洞的审计员相比初始开发软件的程序员要更加熟悉语言和代码库。在一些案例中，这种级别的理解可以简单通过关注可选编译器警告或通过第三方分析工具帮助跟踪常见编程错误而达到。计算机安全等同于计算机精通。对你的目标没有严苛的理解，也就没有抵御攻击的希望。

- Essential C——C语言编程入门
- TAOSSA Chapter 6: C Language Issues——强烈推荐阅读
- Integer Overflow
- Wireshark Security——许多漏洞的例子
- Gera's Insecure Programming by Example——小型漏洞C程序的例子

2.2、二进制审计

现在你已经深入到原生层，这是你撕扯下所有遮掩后的软件。今天我们所关注的原生代码形式是Intel X86下的32位代码。Intel处理器从上世纪80年代开始在个人计算机市场有着强劲的表现，现在支配着桌面和服务器市场。理解这些指令集可以帮助你以内部视角看到程序每天是如何运行的，也可以在你遇到诸如ARM、MIPS、PowerPC和SPARC等其他指令集时提供一种参考。

这部分内容我们将要逐渐熟悉原生层和开发策略，以理解、分析和解释本地代码。在该部分结束时你应该能够完成一项“逆向编译”——从汇编分段到高级语言状态——以及处理过程、派生意义和程序员意图。

课程

学习X86初看起来令人生畏且需要一些专业性的学习才能掌握。我们建议阅读《深入理解计算机系统》的第三章学习C程序是怎样编译成机器指令的。当你有了一些基础的、这种过程的应用知识，就在手边随时备着像弗吉尼亚大学x86汇编指南这样的参考指南。我们还发现了来自Quinn Liu的系列视频作为快速介绍。

- 《深入理解计算机系统》第三章: Machine-Level Representation of Programs
- x86 Assembly Guide
- Introduction to x86 Assembly

挑战工坊

下面的程序都是“二进制炸弹”。逆向工程这些Linux程序并确定输入序列就可以“拆除“炸弹。炸弹的每个连接层关注于原生代码的不同层面。例如，CMU实验室中的程序（CMU Binary Bomb Lab）中你会看到不同数据结构（链表、树）以及控制流结构（切换、循环）在原生代码层面怎样表现的。在逆向这些程序时你可以发现将程序执行流程转换为C或者其他高级语言的有用之处。

你应该将目标聚焦于解决这两个实验室程序的八个段。CMU炸弹程序有一个隐藏段，RPI炸弹程序有一个段包含有内存错误，你能找到并解决么？

- CMU Binary Bomb Lab
- RPI Binary Bomb Lab

工具

处理原生代码的两个至关重要的工具是调试器和反汇编器。我们建议你熟悉下行业标准反汇编工具：IDA Pro。IDA会分割代码为独立的块以对应程序源码的定义函数。每个函数进一步被分割为修改控制流的指令定义的“基础块”。这样很容易一眼识别循环、条件和其他控制流指

令。

调试器允许你与设置了断点的运行中代码进行交互和状态检查，以及内存检查和寄存器内容查看。你会发现如果你的输入没有产生预期的结果，这些查看功能是很有用的，但是一些程序会使用反调试技术在调试时改变程序行为。对于大多数Linux系统来说GNU调试器（gdb）是标准的调试工具。gdb可以通过你所用Linux版本的软件包管理器获得。

- IDA Pro Demo
- gdb

资源

已经有许多好的资源用来学习x86汇编和CTF题目中的技巧。除了以上资源，x86维基手册和AMD指令集帮助都是更加完整的参考供你参考（我们发现AMD帮助手册没有Intel帮助手册那么吓人）。

- AMD64程序员帮助手册: General-Purpose and System Instructions
- x86 Assembly Wikibook
- Computer Systems: A Programmer's Perspective (《深入理解计算机系统》)

一些逆向工程工具使用起来和汇编语言自身一样复杂。下面列出的是常用的命令行工具的命令表单：

- gdb Quick Reference
- IDA Quick Reference
- WinDBG x86 Cheat Sheet

最后，许多CTF挑战会使用反调试技术和反汇编技术隐藏或混淆目标。上面的炸弹程序就使用了其中的几种技术，但是你可能想要更全面的参考资料。

- Linux anti-debugging techniques
- The "Ultimate" Anti-Debugging Reference

2.3、Web应用审计

欢迎来到Web渗透部分，在这个部分中将会熟悉Web应用中发现的常见漏洞。在该部分结束时你应该能够使用各种测试方法和源码级别审计识别基于Web的常见应用漏洞。课程资源中将会给出对挑战工坊资源进行成功审计的所有工具。

课程

- Web Hacking Part I
- Web Hacking Part II

挑战工坊

为了锻炼你的技能，我们建议你实践一遍Damn Vulnerable Web App（DVWA）和Sibria Exploit Kit的漏洞发掘与利用。DVWA是基于PHP并汇总了各类漏洞的一套测试环境，在其中能够看到Web应用中许多常见的错误。Siberia Exploit Kit是一个被许多犯罪份子用来完成大量攻击的“犯罪套件”，它包括了一个浏览器利用包和一个用来管理受害主机的控制面板。Siberia包含的几种基于POST的身份认证漏洞允许攻击者获得管理员权限并接管服务器所在的主机。

下载并在VMware虚拟机中运行OWASP Broken Web Apps开始此次挑战工坊。BWA包含许多用来安全测试的Web应用，其中包括DVWA。如果你搞定了DVWA，那么放轻松再试试其他Web应用漏洞！尝试审计Siberia的源码来找寻漏洞，要将注意力集中在PHP输入的代码上。

- OWASP Broken Web Apps
- Siberia Crimeware Pack (口令: infected)

工具

Burp Suite是一个用来安全测试的本地HTTP代理。Burp Suite是针对Web渗透测试人员开发的，并将许多常见功能集成在一个GUI界面中。免费版本的可用功能足以完成上面的挑战和许多其他Web安全挑战。

- Burp Suite

资源

许多简单的测试方法和常见的Web应用漏洞都可以在该部分的挑战题中遇到。在做这些挑战题前请先确定你理解了HTTP、HTML和PHP的基础知识。

- OWASP Top 10 Tools and Tactics
- The Tangled Web: Chapter 3
- PHP Primer

三、漏洞利用程序的构建

3.1、二进制的漏洞利用（1）

二进制的漏洞利用是破坏编译程序的过程，令程序违反自身的可信边界从而有利于你——攻击者。本部分中我们将聚焦于内存错误。通过利用漏洞来制造软件内存错误，我们可以用某种方式重写恶意程序静态数据，从而提升特定程序的权限（像远程桌面服务器）或通过劫持控制流完成任意操作和运行我们所用的代码。

如果你尝试在已编译的C程序中找bug，知晓你要找的东西是很重要的。从认识你发送的数据被程序用在什么地方开始，如果你的数据被存储在一个缓冲区中，要注意到它们的大小。编写没有错误的C程序是非常难的，CERT C Coding Standard手册汇总了许多错误出现的方式。对commonly misused APIs稍加注意是可以加快了解的捷径。

一旦一个漏洞被确认，它就可以被用来威胁程序的完整性，然而，有各种不同的方式可以达到这个目标。对于像Web服务器这样的程序，获取另一个用户的信息可能是最终目标。另一方面，修改你的权限可能是有用的，比如修改一个本地用户权限为管理员。

课程

第一套课程是《Memory Corruption 101》，提供了Windows环境下缓冲区溢出利用一步一步的解释和相关背景。第二套课程《Memory Corruption 102》，涵盖了更多高级主题，包括Web浏览器漏洞利用。这两套课程都是针对Windows的例子，但技术和过程可以用于使用x86指令集的其他操作系统。记住，当你处理UNIX/Linux二进制时，函数名称和有时候的调用约定是不一样的。

- Memory Corruption 101
- Memory Corruption 102

工具

我们建议使用GDB来调试该部分的挑战题，因为它们都是在32位Linux下编译的，然而，GDB更适合用来调试源代码，而不是没有标识符和调试信息的纯粹二进制程序。诸如gdbinit、peda和voltron可以使gdb在调试无源码的程序时更有用。我们建议在你的home目录下创建一个至少包含以下命令的.gdbinit文件：

```
set disassembly-flavor intel

set follow-fork-mode child
```

挑战工坊

为了运行这些挑战题，你需要安装Ubuntu 14.01（32-bit）虚拟机。我们建议使用VMware Player，因为它免费且支持性很好。在你运行虚拟机后，打开终端并运行sudo apt-get install socat来安装socat。

在此次挑战工坊中共有三道挑战题，当你在git中clone该手册的repository后，每道题都包含在其目录中，每道题的最终目标是操纵执行流以读取flag。对于每道题，请尝试将反汇编转换为C代码。在尝试之后，你可以通过查看提供的实际C源码来确认你的猜测，然后，尝试利用漏洞来读取flag。

挑战题：**Easy**

确认目录中easy程序的flag。当你执行easy后，它会在12346端口监听指令。

挑战题：**Social**

和easy类似，确认目录中social程序的flag和作为social程序的host.sh。当你执行social后，它将在12347端口监听指令。

资源

- Using GDB to Develop Exploits – A Basic Run Through
- Exploiting Format String Vulnerabilities
- Low-level Software Security: Attacks and Defenses

3.2、二进制的漏洞利用（2）

在该部分内容中，我们继续可利用漏洞的本地应用检查之路，并关注使用返回导向编程（ROP）来达到此目的。ROP是在代码结尾的返回指令中整合现有可执行片段的过程。通过创建这些“玩意儿”地址链可以在不引入任何新代码的情况下写新程序。

记住，在可利用程序的漏洞识别方法上你需要灵活应变。有时候有必要在漏洞利用开发过程中对一个漏洞多次利用。有时，你可能仅想用ROP来让你的shellcode执行，其他情况下，你可能想在ROP中完整写一个攻击载荷。偶尔，内存布局能使非常规的漏洞利用方法可行，例如，你可曾考虑过用ROP来构造一个不受控的格式化字符串漏洞？

课程

本部分的课程将讨论返回导向编程（ROP）和绕过数据不可执行保护的代码重用。这些在漏洞利用细节上会更具体，且基于上部分所讨论的内容。

- Return Oriented Exploitation
- Payload already inside data re-use for ROP exploits（特指Linux）

挑战工坊

和前一个部分的挑战一样，在你clone repository后文件夹中有两个可执行文件。每个程序的漏洞利用都需要使用返回导向编程以读取flag。此次的挑战题没有提供源代码的访问。你需要对二进制程序进行逆向工程来发掘漏洞，并需要将漏洞利用的技术。同样，请使用相同的Ubuntu 14.04（32-bit）虚拟机。

挑战题：**brute_cookie**

运行bc程序，它会监听12345端口。

挑战题：**space**

运行作为space程序的host.sh，它会监听12348端口。

挑战题：**rop_mixer**

运行作为rop_mixer程序的host.sh，它会监听12349端口。

工具

参考前面所提到的工具。如果你还没有准备，你需要熟悉一下*NIX的binutils套件。像readelf、strings、objdump、objcopy和nm都是常用的有用工具。请使用软件包管理器和帮助页面安装和阅读它们的使用。

有几个现有的工具专门用来创建可重用代码的漏洞，它们比一般的反汇编器更加专业，因为它们会寻找合适的可执行代码段作为ROP目标（在指令、.rodata等中）。

- RP
- ROPGadget
- BISC——适用于课程的简单工具

资源

- x86-64 buffer overflow exploits and the borrowed code chunks exploitation technique
- Surgically returning to randomized lib(c)
- Extensive security reference
- Dartmouth College: Useful Security and Privacy links
- Corelan Team Blog

3.3、Web应用的漏洞利用

该部分承接前面的Web应用审计部分。在该部分中我们将关注于漏洞的利用，在结束时你应该能够熟练地识别和利用OWASP Top 10。

课程

前面的内容中我们已经介绍了Web安全的基础部分，所以现在在该部分中我们可以更深一步到一些能够获得更大效果的合适工具。学习掌握Burp Suite和Chrome开发者工具能够更好的理解和你交互的应用程序。BeEF是一个XSS代理的例子，通读它的源码学习它怎样工作将会受益匪浅。

- Burp Suite Training
- Chrome Dev Tools
- From XSS to reverse shell with BeEF

挑战工坊

你被指派对一个小而糙的Web应用程序Gruyere进行审计。Gruyere是托管于Google的应用，它能够进行许多类型的Web方面漏洞利用练习，包括XSS、SQL注入、CSRF、路径穿越和其他。对于每一个挑战你都能够找到提示、漏洞和进行漏洞修复的方法。

参考

- Google Chrome Console
- OWASP Top 10 Tools and Tactics
- The Tangled Web: Chapter 3
- PHP Primer

工具

SQLMap和BeEF都是强大且有趣的工具，但是对于挑战题基本上用不着。如果你坚持要使用BeEF，请不要尝试拦截进行应用审计的其他用户 。

- Burp Suite
- SQL Map
- The Browser Exploitation Framework (BeEF)

四、电子取证

五、工具箱的构建

5.1、工具箱的准备

欢迎来到工具箱构建部分。工具箱就是能够让你和你的团队以最有效的方式达成目标的工具集合。工具箱是一个有强有力的效率工具，能够让你在比赛期间花费最短的时间开发漏洞利用并得到最大化的开发回报。

一个好的工具箱是好用且易用的。你应该将能够让团队有效沟通、协同工作、自动化的常用工具和提供比赛情况的软件包含进来。

课程

- Creating a SOC
- Stealth Rootkit Development
- Toolsmithing Case Study
- Organizing and Participating in CTF
- RTFn

挑战工坊

创建三个列表。第一个列表用你理想化的功能型工具构成；第二个列表用能够提供你所需要的功能的软件构成；第三个列表从前两个列表中按照功能重要性和支持性进行排序。最后开发那些能够弥补你理想化工具和现有工具不足的软件。

下面是一些你不能忽视的功能：

- 漏洞利用、密钥聚集和提交管理
- 隐秘和安全的载荷或持续驻留方法
- 安全通讯和协同
- 网络/主机环境监测

资源

- Meterpreter Functionality Outline
- IDA Python Overview, IDA Python Download
- NASM Documentation
- Pyershark
- 从Pwnies（Python）和Ronin（Ruby）你可能会找到有用的代码

六、可实施的谍报技术

6.1、案例研究

可实施的谍报技术通常是持续对一个特定目标的挖掘。在进行比赛性的Wargame时你将会几乎将所有经历集中在规避检测并不暴露你的工具集（基础工具、漏洞利用）的风险。

课程

- Post-Exploitation and Operational Security
- A Brief History of CTF and Tradecraft
- Operational Use of Offensive Cyber

挑战工坊

- 评价下面活动中所展现的可实施谍报技术。每个计划都决定了要使用的工具和活动背后的可操作性侧露。
- 在评价谍报技术过程中思考以下一些问题：
- 为什么行动者选择完成/实现X行动/能力？
- 哪里有失误？漏洞的选择还是某种形式的短视？
- 行动/能力有不恰当么？能否弥补可操作性策略的不足？
- 行动者对保护什么最有兴趣？（比如：工具、特征、雇主等等）
- 从攻击者角度每个活动中能学到什么？防御者角度呢？

活动

- APT1
- StuxNet
- Careto

资源

这些是少数公开的关于他们自己谍报技术的例子、团队或组织。下面的提供的AMA是对格外天才的团队怎样行动的惊鸿一瞥。

- Plaid Parlement Of Pwning AMA
- Samurai AMA
- Tradecraft Lectures 8 & 9

七、参与者

该手册的编写建立在许多艰难工作之上，绝大多数难点是在其他地方。未经允许不得重新发布这些文件，因为该手册尚在更新过程中，我们非常感谢和感激大家的支持。

所以，读者们，当你完成一些CTF比赛准备参加更多的比赛，请参与到该手册的编写中来。那些有着雄心壮志的人们或许知道一些需要如此技能的人。

- Andrew Ruef 提供了倡议和PPT。
- Evan Jensen 开发和优化了几乎有所的课程
- Nick Anderson 校对了所有课程并进行了许多改进
- Alex Sotirov 贡献了ROP课程并提供了反馈意见
- Jay Little 重新审阅了二进制漏洞利用部分的内容
- Brandon Edwards 贡献了源码审计课程和newspaper应用
- Marcin W 和Gotham Digital Science 贡献了Web安全课程
- Dino Dai Zovi 贡献了漏洞利用课程的介绍

如果你对这样的课程感兴趣，请参考NYU Poly。他们专注于网络安全且经常通过他们的Hacker in Residence活动与我们合作。

参与进来

如果你想参与该手册的编写，只需简单地将你新的markdown提交到master分支即可，我们可以从那里获取到提交。Gitbook有一个出色的编辑器用来帮助你预览更改。我们一直在找寻新或精妙的内容，所以请给我们提供建议！

Gitbook的使用

《CTF领域指南》是用Gitbook创建的，它是一个用Git和Markdown创建电子书的命令行工具。你可以用Gitbook将《CTF领域指南》导出为PDF格式、一个eBook或一个单独可打印的HTML页面。在安装Gitbook和它的少量插件前请确保你的电脑上安装有NodeJS和npm：

```
npm install gitbook gitbook-plugin-ga gitbook-pdf ebook-convert -g
```

在Gitbook安装后，你可以在书籍目录内运行下面的任一命令：

- 创建一个互动、静态站点：`gitbook build ./myrepo`
- 创建一个独立页面站点：`gitbook build ./myrepo -f page`

- 创建一个PDF文件：`gitbook pdf ./myrepo`（需要gitbook-pdf）

（全文待续）